UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/777,150	02/13/2004	Woo-jong Park	Q79322	4042

23373 7590 09/16/2010

SUGHRUE MION, PLLC
2100 PENNSYLVANIA AVENUE, N.W.
SUITE 800
WASHINGTON, DC 20037

EXAMINER	
AHMED, SALMAN	
ART UNIT	PAPER NUMBER
2476	

NOTIFICATION DATE	DELIVERY MODE
09/16/2010	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

sughrue@sughrue.com
PPROCESSING@SUGHRUE.COM
USPTO@SUGHRUE.COM

Office Action Summary	Application No.	Applicant(s)	
	10/777,150	PARK, WOO-JONG	
	Examiner	Art Unit	
	SALMAN AHMED	2476	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *01 June 2010*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-10* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,2,6 and 7* is/are rejected.

7)☒ Claim(s) *3-5 and 8-10* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 1, 2 and 6-7 are rejected under 35 U.S.C. 103(a) as being unpatentable over Oskouy et al. (US PAT PUB 2007/0147257, hereinafter Oskouy) in view of Kim et al. (US PAT PUB 2002/0078196, hereinafter Kim).

In regards to claim 1, Oskouy teaches *a packet forwarding system (system in figure 3d), comprising: an input unit* (figure 3d, input interface 300) *for inputting first data in units of transmission* (paragraph 0047, Each multifunction multiport includes a line input interface 300 that can include up to 16 input ports for receiving data from up to 16 active streams. Each multi-function multiport is configurable to accommodate streams of various formats and in one implementation supports a 2.4 Gbps (OC-48) full duplex interface. Other input configurations are available with an accumulated bandwidth of up to 2.4 Gbps, including up to 16 streams of OCA3 bandwidth. Other configurations include 12 streams of OC-3 bandwidth and one stream of OC-12 bandwidth); *a packet memory management unit* (figure 3d, 380, 387 and 341 in combination) *for assembling the input first data into packet and loading the packet into a packet memory* (paragraph 0047-0048, Packing queue 380 receives the various input stream data on the input ports and packs the stream data into data words for transfer to segmentation buffer 387 (up to 64 bytes in one implementation) Packing queue 380 includes per stream queues

for accumulating data from each stream. Bit packing queue 381a includes a controller 385a for streaming data words from the bit packing queue to a byte packing queue 381b. Each stream bit packing queue can be sized to hold two or more words of data, and in one implementation, are sized to hold three 64 bit words of data (8 byte data words). Associated with each data word stored in the stream bit packing queues are control flags which are received along with the stream data. In one embodiment, five (5) bits of flag data are stored with each 64 bit data word in each stream bit packing queue. The five flag data bits are pasted with each 64 bit word from the bit packing queue 381a to the byte packing queue 381b), *and reading out a packet header* (paragraph 0057, data words are transferred from byte packing queue 381b to segmentation buffer 387 as they become available. In addition, coincident with the transfer, the first 32 bytes of data packet (four 8 byte data words) are also transferred to L2 header buffer 383 of L2 pattern match decoder 382) *and a pointer of a packet trailer connected to the packet header* (paragraph 0061, Associated with the segmentation buffer is a selectable start read pointer that indicates the location in the segmentation buffer to begin read operations (when reading data from the segmentation engine by packetizer 391). Data bytes are read from byte packing queue 381b and stored in segmentation buffer 387); *a header processing unit* (figure 3d, 382) *for deciding a packet classification and a transmission destination by using the packet header provided from the packet memory management unit* (paragraph 0060, Associated with the L2 header parser is a per stream L2 state queue 419. Per stream state queue 499 stores flags associated with a micro-code starting address, priority (precedence) flag for

the stream, an interface index mapping to one or more logical interfaces, virtual connection stream state information and channel stream state information. The per stream state queue stores information associated with each stream so as to assure continuity in steam processing), *and reporting to the packet memory management unit the pointer of the packet trailer to be connected to the packet header* (paragraph 0066, The offset may not arise exactly on an eight byte boundary necessitating the storage of overflow data. To facilitate block transfers from byte packing queue 381b to segmentation buffer 387 (8 byte blocks), an overflow queue is provided. Segmentation state queue 401 includes a queue sized to contain N-1 bytes of data for each stream, where N is equal to the size of the block transfers from byte packing queue 381b to segmentation buffer 387. Extra bytes that are required to be read from byte packing queue 381b to facilitate the filling of a cell in cell payload queue 388 are stored in segmentation state queue 401); *and an output unit* (figure 3d, 388, 390 and 392 in combination) *for dividing the packet header and the packet trailer into second data in units of transmission* (paragraphs 0065, 0067, the offset information is derived from the L2 packet header. The offset information can be provided in the form of a pointer pointing to a particular location in the data stored in segmentation buffer 387. The pointer can be used to indicate the particular byte in the data transferred from the segmentation buffer that marks the beginning the next layer header. At the beginning of a cell packing operation, cell packetizer 391 first checks to determine if extra bytes for the stream being processed are present in segmentation state queue 401. If so, the extra bytes are loaded first prior to the loading of bytes from the segmentation buffer

387. A cell segmentation engine 400 oversees the transfer of data from segmentation state queue 401 to cell packetize 391. In one implementation, each queue in the segmentation state queue 401 stores up to seven leftover bytes of data that may have been required to be read in order to fill the previous cell associated with the given stream), *receiving the packet header directly from the header processing unit, reading the packet trailer directly from the packet memory of the packet memory management unit based on the reported pointer of the packet trailer to be connected to the packet header* (paragraphs 0062, 0064, 0068, 0078 the header of a cell is generated from data associated with the cell type (indirect cell, direct cell, and data cells) along with header flags extracted during L2 and L3 header processing as described above. In one implementation, the position in the cell header queue 389 that the header is written to is derived from the location in the cell payload queue 388. The parallelism of the addressing for headers and cell data portions simplifies cell processing. The segmentation buffer provides temporary storage of the data words transferred from the byte packing queue while the L2 processing completes. Transfer from segmentation buffer 387 by cell packetizer 391 to cell payload queue 388 is initiated upon completion of the L2 processing and delivery of the offset information to cell packetizer 391. Cell packetizer 391 is an engine that services cell segmentation buffer 387 providing a cell sized amount of data to cell payload queue 388. Each cell includes a data portion and a header portion. In one implementation, cell packetizer 391 transfers 64 bytes of data to cell payload queue 388. The position in cell payload queue 388 to which the cell is written is controlled by buffer pool manager 393. Cell dispatcher 385b receives

feedback from buffer pool manager 393 as entries are extracted from Cell payload queue 388. A slot must be available in cell payload queue 388 prior to the packetizing of the cell data and extraction from byte packing queue 381b. Buffer pool manager includes a pointer that indicates the next available cell in the cell payload queue that can be written to by cell packetizer 391. As each cell is written into cell payload queue 388 an associated header is written into the cell header queue 390. The header of a cell is generated from data associated with the cell type (indirect cell, direct cell, and data cells) along with header flags extracted during L2 and L3 header processing as described above. In one implementation, the position in the cell header queue 389 that the header is written to is derived from the location in the cell payload queue 388. The parallelism of the addressing for headers and cell data portions simplifies cell processing), *and outputting the second data to a channel* (paragraphs 0060, 0080, channel stream, Bank engine queue 397 extracts the indicated cell data from payload queue 388 and the associated header information from cell header queue 390. The complete cell is provided as an output to input switch interface 304).

Oskouy does not explicitly teach processing packet as Internet Protocol (IP) packet and inputting first data in units of transmission, assembling the input first data into an Internet Protocol (IP) packet and an output unit for dividing the IP packet header and the IP packet trailer into second data in said units of transmission.

Kim in the same or similar field of endeavor teaches the ATM input port processor 41 includes an input line connecting part 411 for extracting an ATM cell from an ATM transmission line frame, and connecting it to an input line; an ATM cell filtering

part 412 for discriminating a cell required for an IP packet process and a cell required for only an ATM process, among the ATM cells received through the input line connection part 411, and filtering the discriminated cell; an IP reassembling part 413 for reassembling the cells transferred from the ATM cell filtering part 412 in an IP packet type; an IP hop detecting part 414 for searching an IP hop to find out a next-hop of an IP packet reassembled through the IP reassembling part 413; an IP packet queue per output port 415 for storing the IP packet completed in a process of an IP header for the next-hop through the IP hop detecting part 414, per output port; an ATM cell division part 416 for performing a sequential access to a queue per port through the IP packet queue 415 and executing a division into ATM cells; and an input queue controlling part 418 for performing an alternate access to the ATM cell queue per output port 415 for storing, per output port, the cell transferred from the ATM cell filtering part 412, the ATM cell division part 415 and the ATM cell queue per output port 417, and transferring the ATM cell to the cell-based switch fabric 42 (paragraph 0030).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to modify Oskouy's system/method the steps of processing packet as Internet Protocol (IP) packet and inputting first data in units of transmission, assembling the input first data into an Internet Protocol (IP) packet and an output unit for dividing the IP packet header and the IP packet trailer into second data in said units of transmission as suggested by Kim. The motivation is that (as suggested by paragraph 0001 of Kim) such method enables an IP packet forwarding dispersion processing apparatus and method for effectively dispersion-processing an IP (Internet Protocol)

forwarding function and a routing information base on the basis of a QoS (Quality of Service) within a system. Known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces/market place incentives if the variations are predictable to one of ordinary skill in the art.

In regards to claim 6, Oskouy teaches a packet forwarding method, comprising: *inputting by an input unit first data in units of transmission* (paragraph 0047, Each multifunction multiport includes a line input interface 300 that can include up to 16 input ports for receiving data from up to 16 active streams. Each multi-function multiport is configurable to accommodate streams of various formats and in one implementation supports a 2.4 Gbps (OC-48) full duplex interface. Other input configurations are available with an accumulated bandwidth of up to 2.4 Gbps, including up to 16 streams of OCA3 bandwidth. Other configurations include 12 streams of OC-3 bandwidth and one stream of OC-12 bandwidth); *a packet memory management step of generating, by a packet memory management unit* (figure 3d, 380, 387 and 341 in combination), *the input first data into packet and loading the packet into a packet memory* (paragraph 0047-0048, Packing queue 380 receives the various input stream data on the input ports and packs the stream data into data words for transfer to segmentation buffer 387 (up to 64 bytes in one implementation) Packing queue 380 includes per stream queues for accumulating data from each stream. Bit packing queue 381a includes a controller 385a for streaming data words from the bit packing queue to a byte packing queue 381b. Each stream bit packing queue can be sized to hold two or more words of data,

and in one implementation, are sized to hold three 64 bit words of data (8 byte data words). Associated with each data word stored in the stream bit packing queues are control flags which are received along with the stream data. In one embodiment, five (5) bits of flag data are stored with each 64 bit data word in each stream bit packing queue. The five flag data bits are pasted with each 64 bit word from the bit packing queue 381a to the byte packing queue 381b), *and reading out and sending packet header* (paragraph 0057, data words are transferred from byte packing queue 381b to segmentation buffer 387 as they become available. In addition, coincident with the transfer, the first 32 bytes of data packet (four 8 byte data words) are also transferred to L2 header buffer 383 of L2 pattern match decoder 382) *and a pointer of packet trailer connected to the packet header* (paragraph 0061, Associated with the segmentation buffer is a selectable start read pointer that indicates the location in the segmentation buffer to begin read operations (when reading data from the segmentation engine by packetizer 391). Data bytes are read from byte packing queue 381b and stored in segmentation buffer 387); *a header processing step of deciding, by a header processing unit* (figure 3d, 382), *a packet classification and a transmission destination by using the packet header provided from the packet memory management unit* (paragraph 0060, Associated with the L2 header parser is a per stream L2 state queue 419. Per stream state queue 499 stores flags associated with a micro-code starting address, priority (precedence) flag for the stream, an interface index mapping to one or more logical interfaces, virtual connection stream state information and channel stream state information. The per stream state queue stores information associated with each

stream so as to assure continuity in steam processing), *and reporting to the packet memory management unit the pointer of the packet trailer to be connected to the packet header* (paragraph 0066, The offset may not arise exactly on an eight byte boundary necessitating the storage of overflow data. To facilitate block transfers from byte packing queue 381b to segmentation buffer 387 (8 byte blocks), an overflow queue is provided. Segmentation state queue 401 includes a queue sized to contain N-1 bytes of data for each stream, where N is equal to the size of the block transfers from byte packing queue 381b to segmentation buffer 387. Extra bytes that are required to be read from byte packing queue 381b to facilitate the filling of a cell in cell payload queue 388 are stored in segmentation state queue 401); *and an output step for dividing, by an output unit* (figure 3d, 388, 390 and 392 in combination), *the packet header and the packet trailer into second data in said units of transmission* (paragraphs 0065, 0067, the offset information is derived from the L2 packet header. The offset information can be provided in the form of a pointer pointing to a particular location in the data stored in segmentation buffer 387. The pointer can be used to indicate the particular byte in the data transferred from the segmentation buffer that marks the beginning the next layer header. At the beginning of a cell packing operation, cell packetizer 391 first checks to determine if extra bytes for the stream being processed are present in segmentation state queue 401. If so, the extra bytes are loaded first prior to the loading of bytes from the segmentation buffer 387. A cell segmentation engine 400 oversees the transfer of data from segmentation state queue 401 to cell packetize 391. In one implementation, each queue in the segmentation state queue 401 stores up to seven leftover bytes of

data that may have been required to be read in order to fill the previous cell associated with the given stream), *for receiving the packet header directly from the header processing unit, for reading the packet trailer directly from the packet memory of the packet memory management unit based on the reported pointer of the packet trailer to be connected to the packet header* (paragraphs 0062, 0064, 0068, 0078 the header of a cell is generated from data associated with the cell type (indirect cell, direct cell, and data cells) along with header flags extracted during L2 and L3 header processing as described above. In one implementation, the position in the cell header queue 389 that the header is written to is derived from the location in the cell payload queue 388. The parallelism of the addressing for headers and cell data portions simplifies cell processing. The segmentation buffer provides temporary storage of the data words transferred from the byte packing queue while the L2 processing completes. Transfer from segmentation buffer 387 by cell packetizer 391 to cell payload queue 388 is initiated upon completion of the L2 processing and delivery of the offset information to cell packetizer 391. Cell packetizer 391 is an engine that services cell segmentation buffer 387 providing a cell sized amount of data to cell payload queue 388. Each cell includes a data portion and a header portion. In one implementation, cell packetizer 391 transfers 64 bytes of data to cell payload queue 388. The position in cell payload queue 388 to which the cell is written is controlled by buffer pool manager 393. Cell dispatcher 385b receives feedback from buffer pool manager 393 as entries are extracted from Cell payload queue 388. A slot must be available in cell payload queue 388 prior to the packetizing of the cell data and extraction from byte packing queue 381b. Buffer pool

manager includes a pointer that indicates the next available cell in the cell payload queue that can be written to by cell packetizer 391. As each cell is written into cell payload queue 388 an associated header is written into the cell header queue 390. The header of a cell is generated from data associated with the cell type (indirect cell, direct cell, and data cells) along with header flags extracted during L2 and L3 header processing as described above. In one implementation, the position in the cell header queue 389 that the header is written to is derived from the location in the cell payload queue 388. The parallelism of the addressing for headers and cell data portions simplifies cell processing), *and outputting the second data to a channel* (paragraphs 0060, 0080, channel stream, Bank engine queue 397 extracts the indicated cell data from payload queue 388 and the associated header information from cell header queue 390. The complete cell is provided as an output to input switch interface 304).

Oskouy does not explicitly teach processing packet as Internet Protocol (IP) packet and inputting first data in units of transmission, assembling the input first data into an Internet Protocol (IP) packet and an output unit for dividing the IP packet header and the IP packet trailer into second data in said units of transmission.

Kim in the same or similar field of endeavor teaches the ATM input port processor 41 includes an input line connecting part 411 for extracting an ATM cell from an ATM transmission line frame, and connecting it to an input line; an ATM cell filtering part 412 for discriminating a cell required for an IP packet process and a cell required for only an ATM process, among the ATM cells received through the input line connection part 411, and filtering the discriminated cell; an IP reassembling part 413 for

reassembling the cells transferred from the ATM cell filtering part 412 in an IP packet type; an IP hop detecting part 414 for searching an IP hop to find out a next-hop of an IP packet reassembled through the IP reassembling part 413; an IP packet queue per output port 415 for storing the IP packet completed in a process of an IP header for the next-hop through the IP hop detecting part 414, per output port; an ATM cell division part 416 for performing a sequential access to a queue per port through the IP packet queue 415 and executing a division into ATM cells; and an input queue controlling part 418 for performing an alternate access to the ATM cell queue per output port 415 for storing, per output port, the cell transferred from the ATM cell filtering part 412, the ATM cell division part 415 and the ATM cell queue per output port 417, and transferring the ATM cell to the cell-based switch fabric 42 (paragraph 0030).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to modify Oskouy's system/method the steps of processing packet as Internet Protocol (IP) packet and inputting first data in units of transmission, assembling the input first data into an Internet Protocol (IP) packet and an output unit for dividing the IP packet header and the IP packet trailer into second data in said units of transmission as suggested by Kim. The motivation is that (as suggested by paragraph 0001 of Kim) such method enables an IP packet forwarding dispersion processing apparatus and method for effectively dispersion-processing an IP (Internet Protocol) forwarding function and a routing information base on the basis of a QoS (Quality of Service) within a system. Known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other

market forces/market place incentives if the variations are predictable to one of ordinary skill in the art.

In regards to claims 2 and 7, Oskouy teaches the packet memory management unit includes: a packet generator for generating the packet from the input first data (paragraph 0048, bit packing queue 381a includes a controller 385a for streaming data words from the bit packing queue to a byte packing queue 381b); the packet memory comprising plural buffers (figure 3d, element 381b, queues 0-15) loading the packet (paragraph 0048, bit packing queue 381a includes a controller 385a for streaming data words from the bit packing queue to a byte packing queue 381b), and the plural buffers storing buffer attribute information (paragraph 0048, Each stream bit packing queue can be sized to hold two or more words of data, and in one implementation, are sized to hold three 64 bit words of data (8 byte data words). Associated with each data word stored in the stream bit packing queues are control flags which are received along with the stream data. In one embodiment, five (5) bits of flag data are stored with each 64 bit data word in each stream bit packing queue. The five flag data bits are pasted with each 64 bit word from the bit packing queue 381a to the byte packing queue 381b) and the pointer of the packet trailer connected to the packet header (paragraph 0049, in one embodiment the control flags include an end of packet without error flag (1 bit), a end of packet with error flag (1 bit), and a last byte pointer (3 bits). The size of the last byte pointer indicates the last byte in the eight (8) byte word of data that contains data in the transfers between the bit packing queue 381a and byte packing queue 381b); a transmission header queue for loading a pointer of the packet header corresponding to

a transmission order of the packet (paragraphs 0064, 0065, Cell packetizer 391 is an engine that services cell segmentation buffer 387 providing a cell sized amount of data to cell payload queue 388. As described above, cells are the preferred storage mechanism for storing a fixed portion of a packet. Each cell includes a data portion and a header portion. In one implementation, cell packetizer 391 transfers 64 bytes of data to cell payload queue 388. Cell packetizer receives as an input cell data from segmentation buffer 387 and from the segmentation state queue 401 as well as offset information from L2 pattern match decoder 382. The operation of the segmentation state queue 401 is described in detail below. As described above, the offset information is derived from the L2 packet header. The offset information can be provided in the form of a pointer pointing to a particular location in the data stored in segmentation buffer 387. The pointer can be used to indicate the particular byte in the data transferred from the segmentation buffer that marks the beginning the next layer header); and a controller for reading from the packet memory the packet header and the pointer of the packet trailer connected to the packet header, according to the transmission order determined by the transmission header queue, and transmitting the pointer of the packet trailer and the packet header to the header processing unit (paragraphs 0064, 0065, 0068, 0069 Cell packetizer 391 is an engine that services cell segmentation buffer 387 providing a cell sized amount of data to cell payload queue 388. As described above, cells are the preferred storage mechanism for storing a fixed portion of a packet. Each cell includes a data portion and a header portion. In one implementation, cell packetizer 391 transfers 64 bytes of data to cell payload queue

388. Cell packetizer receives as an input cell data from segmentation buffer 387 and from the segmentation state queue 401 as well as offset information from L2 pattern match decoder 382. The operation of the segmentation state queue 401 is described in detail below. As described above, the offset information is derived from the L2 packet header. The offset information can be provided in the form of a pointer pointing to a particular location in the data stored in segmentation buffer 387. The pointer can be used to indicate the particular byte in the data transferred from the segmentation buffer that marks the beginning the next layer header. Cell dispatcher 385b receives feedback from buffer pool manager 393 as entries are extracted from Cell payload queue 388. A slot must be available in cell payload queue 388 prior to the packetizing of the cell data and extraction from byte packing queue 381b. Buffer pool manager includes a pointer that indicates the next available cell in the cell payload queue that can be written to by cell packetizer 391. As each cell is written into cell payload queue 388 an associated header is written into the cell header queue 390. Ordering of cells at the stream level is accomplished through use of cell state queue 402. Cells in the same stream must be extracted from cell payload queue 388 in sequential order. Ordering is achieved by writing a code to a per stream queue 388 as each cell is transferred into cell payload queue 388. More specifically, for each write of a data portion of a cell by cell packetizer 391 into cell payload queue 388, cell segmentation engine 400 writes a code that is stored in cell state queue 402. Cell state queue 402 includes a queue for each stream in the input. As each cell is written to cell payload queue 388, a code including address and state information is written to cell state queue 402. The address information

includes a pointer to the location of the data portion of the cell in cell payload queue 388. The state information includes information indicating whether the cell is the first cell, middle cell or last cell in a packet. In one implementation, the code is 10 bits in length and includes two (2) bits indicating whether the cell is a first cell, middle cell or last cell as well as eight (8) bits of address information).

Oskouy does not explicitly teach processing packet as Internet Protocol (IP) packet and inputting first data in units of transmission, assembling the input first data into an Internet Protocol (IP) packet and an output unit for dividing the IP packet header and the IP packet trailer into second data in said units of transmission.

Kim in the same or similar field of endeavor teaches the ATM input port processor 41 includes an input line connecting part 411 for extracting an ATM cell from an ATM transmission line frame, and connecting it to an input line; an ATM cell filtering part 412 for discriminating a cell required for an IP packet process and a cell required for only an ATM process, among the ATM cells received through the input line connection part 411, and filtering the discriminated cell; an IP reassembling part 413 for reassembling the cells transferred from the ATM cell filtering part 412 in an IP packet type; an IP hop detecting part 414 for searching an IP hop to find out a next-hop of an IP packet reassembled through the IP reassembling part 413; an IP packet queue per output port 415 for storing the IP packet completed in a process of an IP header for the next-hop through the IP hop detecting part 414, per output port; an ATM cell division part 416 for performing a sequential access to a queue per port through the IP packet queue 415 and executing a division into ATM cells; and an input queue controlling part

418 for performing an alternate access to the ATM cell queue per output port 415 for storing, per output port, the cell transferred from the ATM cell filtering part 412, the ATM cell division part 415 and the ATM cell queue per output port 417, and transferring the ATM cell to the cell-based switch fabric 42 (paragraph 0030).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to modify Oskouy's system/method the steps of processing packet as Internet Protocol (IP) packet and inputting first data in units of transmission, assembling the input first data into an Internet Protocol (IP) packet and an output unit for dividing the IP packet header and the IP packet trailer into second data in said units of transmission as suggested by Kim. The motivation is that (as suggested by paragraph 0001 of Kim) such method enables an IP packet forwarding dispersion processing apparatus and method for effectively dispersion-processing an IP (Internet Protocol) forwarding function and a routing information base on the basis of a QoS (Quality of Service) within a system. Known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces/market place incentives if the variations are predictable to one of ordinary skill in the art.

Allowable Subject Matter

3. Claims 3-5 and 8-10 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Response to Arguments

Applicant's arguments see pages 7-10 of the Remarks section, filed 6/1/2010, with respect to the rejections of the claims have been fully considered.

Applicant argues that (see page 8) *the Examiner alleges that the combination of packeting queue 380 and cell segmentation engine 386 of Oskouy corresponds to the claimed packet memory management unit. Additionally, the Examiner also alleges that the L2 pattern match decoder 382 and bank spray engine 392 of Oskouy corresponds to the claimed header processing unit and output unit. Assuming arguendo that above correspondences are held true, Applicant submits that Oskouy fails to disclose or suggest at least "an output unit for...receiving the IP packet header directly from the header processing unit, reading the IP packet trailer directly from the packet memory of the packet memory management unit". In other words, Oskouy fails to disclose or suggest the bank spray engine 392 for receiving the IP packet header directly from the L2 pattern match decoder 382 and reading the IP packet trailer directly from the combination of packeting queue 380 and cell segmentation engine 386.*

However, Examiner respectfully disagrees with Applicant's assertion. Oskouy teacehes *a packet memory management unit* (figure 3d, 380, 387 and 391 in combination), *a header processing unit* (figure 3d, 382) and *an output unit* (figure 3d, 388, 390 and 392 in combination); not the elements that the Applicant alleges. Therefore Oskouy teaches receiving the packet header directly from the header processing unit, reading the packet trailer directly from the packet memory of the packet memory management unit based on the reported pointer of the packet trailer to be connected to the packet header (paragraphs 0062, 0064, 0068, 0078 the header of a

cell is generated from data associated with the cell type (indirect cell, direct cell, and data cells) along with header flags extracted during L2 and L3 header processing as described above. In one implementation, the position in the cell header queue 389 that the header is written to is derived from the location in the cell payload queue 388. The parallelism of the addressing for headers and cell data portions simplifies cell processing. The segmentation buffer provides temporary storage of the data words transferred from the byte packing queue while the L2 processing completes. Transfer from segmentation buffer 387 by cell packetizer 391 to cell payload queue 388 is initiated upon completion of the L2 processing and delivery of the offset information to cell packetizer 391. Cell packetizer 391 is an engine that services cell segmentation buffer 387 providing a cell sized amount of data to cell payload queue 388. Each cell includes a data portion and a header portion. In one implementation, cell packetizer 391 transfers 64 bytes of data to cell payload queue 388. The position in cell payload queue 388 to which the cell is written is controlled by buffer pool manager 393. Cell dispatcher 385b receives feedback from buffer pool manager 393 as entries are extracted from Cell payload queue 388. A slot must be available in cell payload queue 388 prior to the packetizing of the cell data and extraction from byte packing queue 381b. Buffer pool manager includes a pointer that indicates the next available cell in the cell payload queue that can be written to by cell packetizer 391. As each cell is written into cell payload queue 388 an associated header is written into the cell header queue 390. The header of a cell is generated from data associated with the cell type (indirect cell, direct cell, and data cells) along with header flags extracted during L2 and L3 header

processing as described above. In one implementation, the position in the cell header queue 389 that the header is written to is derived from the location in the cell payload queue 388. The parallelism of the addressing for headers and cell data portions simplifies cell processing).

Therefore, claims 1-10 stand rejected.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to SALMAN AHMED whose telephone number is (571)272-8307. The examiner can normally be reached on 9:00 am - 5:30 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Ayaz Sheikh can be reached on (571)272-3795. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Salman Ahmed/

Primary Examiner, Art Unit 2476